**Lab Steps**

Task 1: Sign in to AWS Management Console

1. Click on the  button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

* Leave the **Account ID** as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
* Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username** and **Password** in AWS Console and click on the **Sign in** button.

1. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1**.

Task 2: Setup Visual Studio Code

1. Open the Visual Studio Code.
2. If you have already installed and using the Visual Studio code, open a new window.
3. A new window will open a new file and release notes page (only if you have installed or updated Visual Studio Code recently). Close the Release notes tab.
4. Open terminal by selecting View from the Menu bar and choose Terminal.
5. It may take up to 2 minutes to open the terminal window.
6. Once the terminal is ready, let us navigate to the Desktop.

cd Desktop

1. Create a new folder by running the below command:

mkdir task\_10101\_aurora

1. Change your present working directory to use the newly created folder by running the command:

cd task\_10101\_aurora

1. Get the location of the present working directory by running the below command:

pwd

1. Note down the location, as you will open the same in the next steps.
2. Now click on the first icon Explorer present on the left sidebar.
3. Click on the button called **Open folder** and navigate to the location of folder **task\_10101\_aurora.**
4. Visual Studio code is now ready to use.

Task 3: Create a variables file

1. To create a variables file, expand the folder **task\_10101\_aurora** and click on the **New File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note: Don't change the location of the new file, keep it default, i.e. inside the task\_10101\_aurora folder.**
4. Paste the below contents in **variables.tf** file.

|  |
| --- |
| variable "access\_key" {      description = "Access key to AWS console"  }  variable "secret\_key" {      description = "Secret key to AWS console"  }  variable "region" {      description = "AWS region"  } |

1. In the above content, you are declaring a variable called, access\_key, secret\_key, and region with a short description of all 3.
2. After pasting the above contents, save the file by pressing **Ctrl + S**.
3. Now expand the folder **task\_10101\_aurora** and click on the **New File** icon to add the file.
4. Name the file as **terraform.tfvars** and press **Enter** to save it.
5. Paste the below content into the **terraform.tfvars** file.

|  |
| --- |
| region = "us-east-1"  access\_key = "<YOUR\_ACCESS\_KEY>"  secret\_key = "<YOUR\_SECRET\_KEY>" |

1. In the above code, you are defining the dynamic values of variables declared earlier.
2. Replace the values of access\_key and secret\_key by copying from the lab page.
3. After replacing the values of access\_key and secret\_key, save the file by pressing **Ctrl + S**.

Text

Description automatically generated

Task 4: Create a security group for RDS Aurora Instance in main.tf file

 In this task, you will create a main.tf file where you will add details of the provider and resources.

1. To create a **main.tf** file, expand the folder **task\_10101\_aurora** and click on the **New File** icon to add the file.
2. Name the file as **main.tf** and press **Enter** to save it.
3. Paste the below content into the **main.tf** file.

|  |
| --- |
| provider "aws" {      region     = "${var.region}"      access\_key = "${var.access\_key}"      secret\_key = "${var.secret\_key}"  } |

1. In the above code, you are defining the provider as **AWS**.
2. Next, we want to tell Terraform to create a security group for RDS Aurora Instance.
3. To create a security group, Paste the below content into the main.tf file after the provider:

|  |
| --- |
| resource "aws\_security\_group" "allow\_aurora" {    name        = "Aurora\_lab\_sg"    description = "Security group for RDS Aurora"  ingress {      description      = "MYSQL/Aurora"      from\_port        = 3306      to\_port          = 3306      protocol         = "tcp"      cidr\_blocks = ["0.0.0.0/0"]    }  egress {      from\_port        = 0      to\_port          = 0      protocol         = "-1"      cidr\_blocks      = ["0.0.0.0/0"]      ipv6\_cidr\_blocks = ["::/0"]    }  } |

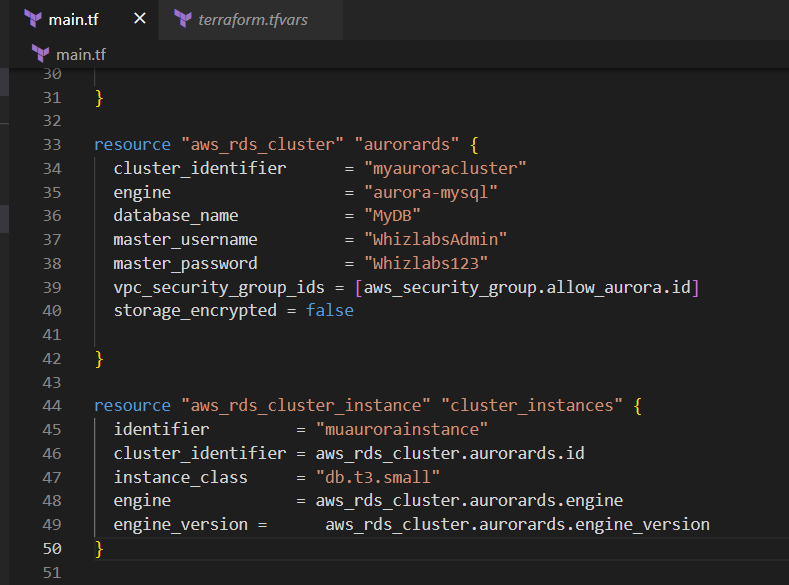
Task 5: Create RDS Database Cluster and Instance in main.tf file

1. First we will create AWS Cluster for the Aurora Instance by defining the cluster identifier, engine, database name, master username, master password, security group and the storage encryption. To add them, paste the following contents in the **main.tf** file:

|  |
| --- |
| resource "aws\_rds\_cluster" "aurorards" {    cluster\_identifier      = "myauroracluster"    engine                  = "aurora-mysql"    database\_name           = "MyDB"    master\_username         = "WhizlabsAdmin"    master\_password         = "Whizlabs123"    vpc\_security\_group\_ids = [aws\_security\_group.allow\_aurora.id]    storage\_encrypted = false    skip\_final\_snapshot   = true  } |

1. To create Aurora instance , paste the following in the **main.tf** file:

|  |
| --- |
| resource "aws\_rds\_cluster\_instance" "cluster\_instances" {    identifier         = "muaurorainstance"    cluster\_identifier = aws\_rds\_cluster.aurorards.id    instance\_class     = "db.t3.small"    engine             = aws\_rds\_cluster.aurorards.engine    engine\_version =      aws\_rds\_cluster.aurorards.engine\_version    } |



1. In the above code, we have defined the identifier, cluster identifier, instance class , engine and engine version for the instance.

Task 6: Create an output file

1. To create an **output.tf** file, expand the folder **task\_10101\_aurora** and click on the **New File** icon to add the file.
2. Name the file as **output.tf** and press **Enter** to save it.
3. Paste the below content into the **output.tf** file.

|  |
| --- |
| output "security\_group\_id" {    value       = aws\_security\_group.allow\_aurora.id  }  output "db\_instance\_endpoint" {    value       = aws\_rds\_cluster\_instance.cluster\_instances.endpoint  } |

1. In the above code , we will extract the security group Id and the DB instance endpoint.

Task 7: Confirm the installation of Terraform by checking the version

1. In the Visual Studio Code, open Terminal by selecting View from the Menu bar and choose Terminal.
2. If you are not in the newly created folder change your present working directory by running the below command.

cd task\_10101\_aurora

1. To confirm the installation of Terraform, run the below command to check the version:

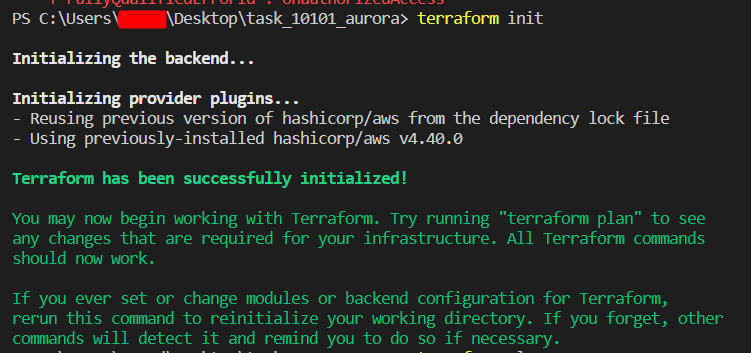
terraform version

1. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

Task 8: Applying terraform configurations

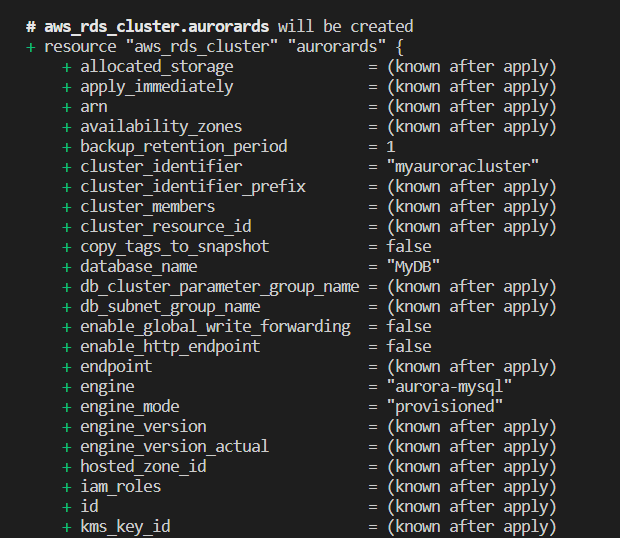
1. Initialize Terraform by running the below command,

terraform init



1. To generate the action plans run the below command,

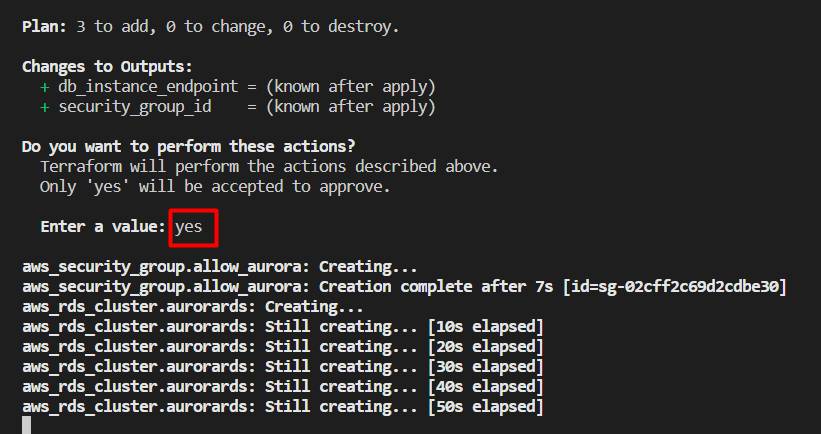
terraform plan



1. To create all the resources declared in main.tf configuration file, run the following command,

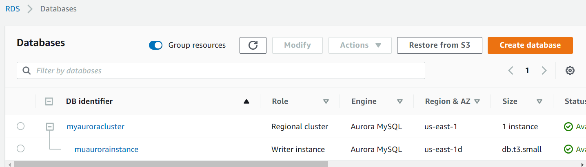
terraform apply

1. Enter **yes** and the resources will be created in 10-15 minutes.



Task 9: Check the resources in the AWS Console

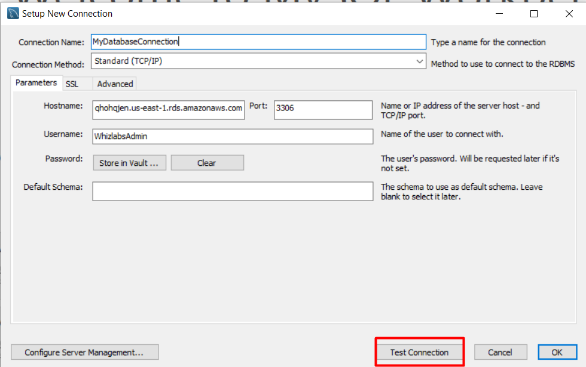
1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **RDS** by clicking on Services on the top, then click on **RDS** in the **Database** section.
3. Click on the **Databases** on the left navigation panel. You can see Aurora Cluster and the Instance created successfully.



Task 10: Testing RDS Connection using the MySQL Workbench

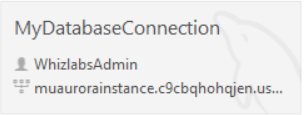
In this task, we will connect to a database on a MySQL DB instance using MySQL monitor commands. One GUI-based application you can use to connect is MySQL workbench, which you have already downloaded and installed based on instructions in the prerequisite section.

1. To connect to a database on a DB instance using MySQL monitor, find the endpoint (DNS name) and port number for your DB Instance.
   * Navigate to **databases** and click on **myaurorainstance**.
   * Under the Connectivity & security section, copy and note the endpoint and port.
     + Endpoint: **Copy the endpoint**
     + For e.g : muaurorainstance.c9cbqhohqjen.us-east-1.rds.amazonaws.com
     + Port: **3306**
     + You need both the endpoint and the port number to connect to the DB instance.
2. Open MySQL Workbench. Click on the plus icon. 
   * Connection Name    : Enter a sample name **MyDatabaseConnection**
   * Host Name         : Enter the **copied endpoint**
   * Port            : **3306**
   * Username        : Enter**WhizlabsAdmin**
   * Password        : Click on Store in Vault and enter password **Whizlabs123**. Click on **ok**.

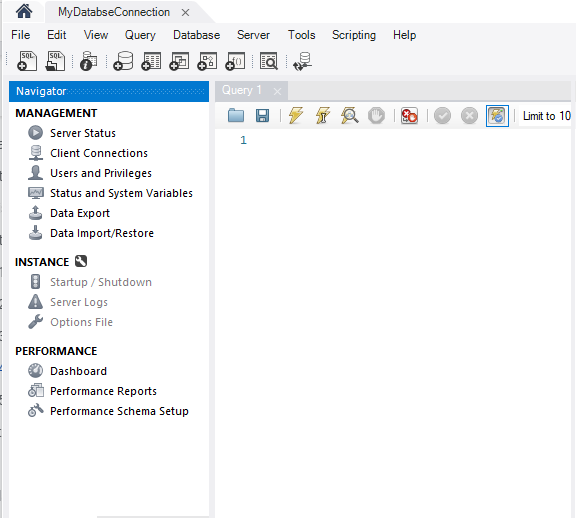


* Click on **Test Connection** to make sure that you are able to connect to the database properly.
* Click on **ok** and ok again to save the connection.

1. A database connection will be created in MySQL Workbench



1. Click on it to open the database. Enter the database password if prompted.
2. After successfully connecting and opening the database, you can create tables and perform various queries over the connected database.



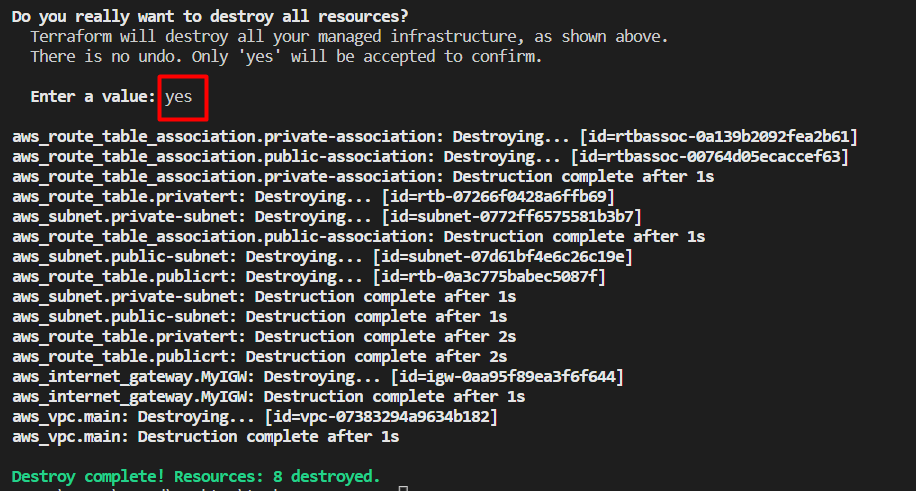
1. Navigate to the Schemas tab to see databases available to start doing database operations. More details on database operations are available [here](https://dev.mysql.com/doc/).

Task 11: Delete AWS Resources

1. To delete the resources, open **Terminal** again.
2. Run the below command to delete all the resources.

terraform destroy

1. Enter **yes** to confirm the deletion. This will take 10-15 minutes to delete.



**Completion and Conclusion**

1. You have set up the Visual Studio Code
2. You have successfully created variables.tf and terraform.tfvars files.
3. You have successfully created a main.tf file.
4. You have executed the terraform configurations commands to create the resources.
5. You have checked the resources created by opening the AWS Console.
6. You have deleted all the resources created.

**End Lab**